

**VISUALIZING**



**THE  
WEB**

# PICTURE CSS3



**Matthew  
David**

# PICTURE CSS3

Matthew David



AMSTERDAM • BOSTON • HEIDELBERG • LONDON • NEW YORK • OXFORD  
PARIS • SAN DIEGO • SAN FRANCISCO • SINGAPORE • SYDNEY • TOKYO

Focal Press is an imprint of Elsevier



Focal Press is an imprint of Elsevier  
30 Corporate Drive, Suite 400, Burlington, MA 01803, USA  
Linacre House, Jordan Hill, Oxford OX2 8DP, UK

© 2010 Elsevier, Inc. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or any information storage and retrieval system, without permission in writing from the publisher. Details on how to seek permission, further information about the Publisher's permissions policies and our arrangements with organizations such as the Copyright Clearance Center and the Copyright Licensing Agency, can be found at our website: [www.elsevier.com/permissions](http://www.elsevier.com/permissions).

This book and the individual contributions contained in it are protected under copyright by the Publisher (other than as may be noted herein).

#### Notices

Knowledge and best practice in this field are constantly changing. As new research and experience broaden our understanding, changes in research methods, professional practices, or medical treatment may become necessary. Practitioners and researchers must always rely on their own experience and knowledge in evaluating and using any information, methods, compounds, or experiments described herein. In using such information or methods they should be mindful of their own safety and the safety of others, including parties for whom they have a professional responsibility.

To the fullest extent of the law, neither the Publisher nor the authors, contributors, or editors, assume any liability for any injury and/or damage to persons or property as a matter of products liability, negligence or otherwise, or from any use or operation of any methods, products, instructions, or ideas contained in the material herein.

ISBN: 978-0-240-81386-8

For information on all Focal Press publications  
visit our website at [www.elsevierdirect.com](http://www.elsevierdirect.com)

Working together to grow  
libraries in developing countries

[www.elsevier.com](http://www.elsevier.com) | [www.bookaid.org](http://www.bookaid.org) | [www.sabre.org](http://www.sabre.org)

ELSEVIER BOOK AID International Sabre Foundation

# CONTENTS

CSS as a Designer's Tools . . . . .	1
Cascading Your Designs. . . . .	3
The Format of CSS. . . . .	7
Modifying Elements with CSS . . . . .	7
Creating Class Styles . . . . .	11
Using Pseudo Class Styles . . . . .	13
Using Pseudo Elements. . . . .	14
Designing Your Web Page with CSS . . . . .	15
Controlling Font Display with CSS. . . . .	15
Embedding Fonts Using CSS3 . . . . .	17
Sizing Fonts with CSS Units of Measurement. . . . .	19
Color Control for Fonts . . . . .	22
Adding Drop Shadow Text Effects. . . . .	23
Additional Font Definitions . . . . .	24
Working with Columns in CSS3 . . . . .	24
Using CSS3 to Control Visual Display . . . . .	26
Positioning Design Elements with CSS . . . . .	26
Increase Control Over Color . . . . .	28
Multiple Background Objects . . . . .	30
Adding Rounded Corners to Layers. . . . .	32
Dazzling Your Audience with CSS3 Animation . . . . .	35
Using Transitions in CSS . . . . .	35
Creating Animation with CSS3 . . . . .	37
Delivering Solutions for the Mobile Market . . . . .	39
What You Have Learned. . . . .	40



CSS gives you the control you need to format content on the screen. Think of CSS as a set of instructions that explain how a document should be presented. Figure 2.2 is the same page from Figure 2.1, formatted with CSS.

CSS has been designed to be easily reused and shared throughout your web site. To this end, it is very easy to switch out design elements. Figure 2.3 is the same page illustrated in Figure 2.1 but with a new CSS design.

You, as a designer, now have much greater freedom in your design. The good news is that working with CSS is not too hard.

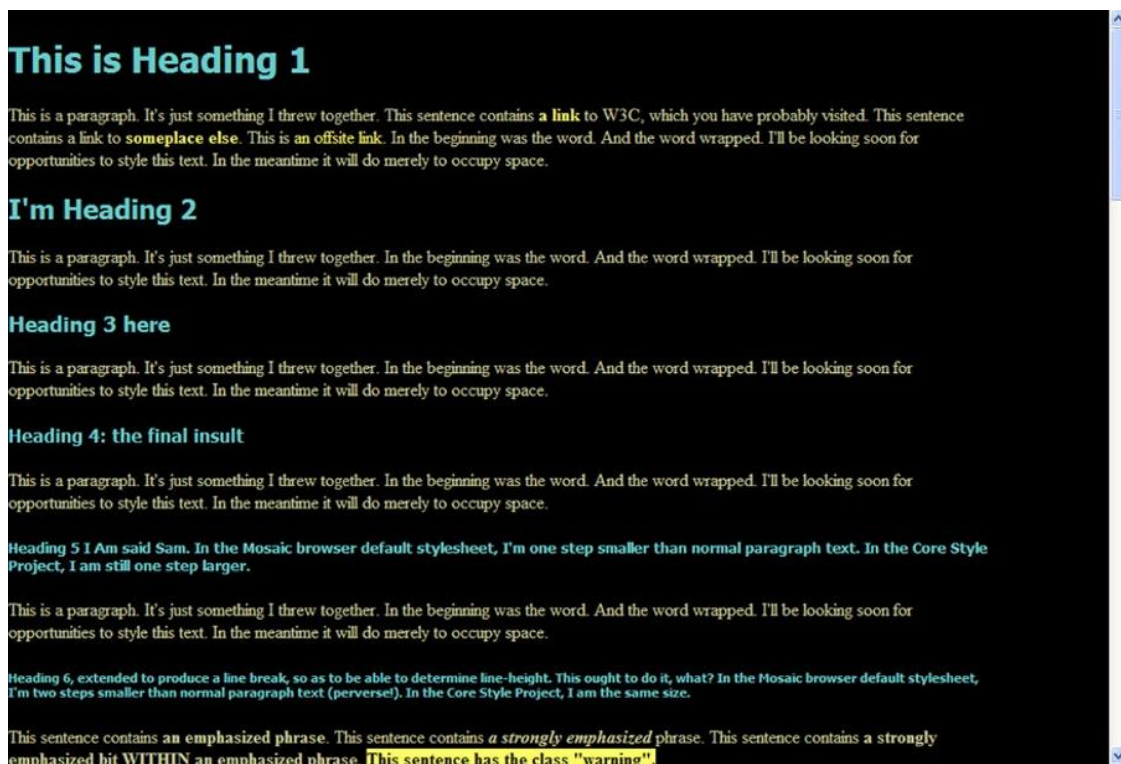


Figure 2.2 CSS is used to format the design of this page.

## This is Heading 1

This is a paragraph. It's just something I threw together. This sentence contains a [link](#) to W3C, which you have probably visited. This sentence contains a link to [someplace else](#). This is an [offsite link](#). In the beginning was the word. And the word wrapped. I'll be looking soon for opportunities to style this text. In the meantime it will do merely to occupy space.

## I'm Heading 2

This is a paragraph. It's just something I threw together. In the beginning was the word. And the word wrapped. I'll be looking soon for opportunities to style this text. In the meantime it will do merely to occupy space.

### Heading 3 here

This is a paragraph. It's just something I threw together. In the beginning was the word. And the word wrapped. I'll be looking soon for opportunities to style this text. In the meantime it will do merely to occupy space.

### Heading 4: the final insult

This is a paragraph. It's just something I threw together. In the beginning was the word. And the word wrapped. I'll be looking soon for opportunities to style this text. In the meantime it will do merely to occupy space.

Heading 5 I Am said Sam. In the Mosaic browser default stylesheet, I'm one step smaller than normal paragraph text. In the Core Style Project, I am still one step larger.

This is a paragraph. It's just something I threw together. In the beginning was the word. And the word wrapped. I'll be looking soon for opportunities to style this text. In the meantime it will do merely to occupy space.

Heading 6, extended to produce a line break, so as to be able to determine line-height. This ought to do it, what? In the Mosaic browser default stylesheet, I'm two steps smaller than normal paragraph text (perverse!). In the Core Style Project, I am the same size.

This sentence contains an emphasized phrase. This sentence contains a *strongly emphasized* phrase. This sentence contains a **strongly emphasized** bit WITHIN an emphasized phrase. **This sentence has the class "warning".**

**Figure 2.3** CSS allows you to easily switch design elements.

## Cascading Your Designs

There are three ways in which you can apply CSS to your HTML content:

- Directly within the HTML element.
- Locally on a web page.
- Externally using a special CSS file to manage your styles for an entire site.

Styles can be applied directly to an HTML element. This is done using the `style` attribute. The following is a section of basic HTML.

```
<h1>
```

```
This is Heading 1
```

```
</h1>
```

```
<p> It's just something I threw together. This
sentence contains <a href="http://www.w3.org/">a
link</a> to W3C, which you have probably visited.
This sentence contains a link to <a href="http://
www.foo.net/">someplace else</a>. This is <a
href="http://www.htmlhelp.com/" class="offsite">an
offsite link</a>. In the beginning was the word.
And the word wrapped. I'll be looking soon for
opportunities to style this text. In the meantime
it will do merely to occupy space.
```

# This is Heading 1

This is a paragraph. It's just something I threw together. This sentence contains [a link](#) to W3C, which you have probably visited. This sentence contains a link to [someplace else](#). This is [an offsite link](#). In the beginning was the word. And the word wrapped. I'll be looking soon for opportunities to style this text. In the meantime it will do merely to occupy space.

## I'm Heading 2

This is a paragraph. It's just something I threw together. In the beginning was the word. And the word wrapped. I'll be looking soon for opportunities to style this text. In the meantime it will do merely to occupy space.

**Figure 2.4** Unformatted HTML code.

```
</p>
<h2>
I'm Heading 2
</h2>
<p>
This is a paragraph. It's just something I
threw together. In the beginning was the word.
And the word wrapped. I'll be looking soon for
opportunities to style this text. In the meantime
it will do merely to occupy space.
</p>
```

Presented in a web page, [Figure 2.4](#) shows how the HTML code looks.

The `style` attribute can now be used to format each element. Take, for instance, the first H1 element; the following style can be applied using CSS.

```
<h1 style="font-family: Verdana, Geneva, Tahoma,
sans-serif;font-size: 24px;color: #FF3300;font-
weight: bolder;">
This is Heading 1
</h1>
```

[Figure 2.5](#) illustrates the change using the new style.

The challenge using the `style` attribute on a specific element is that the style cannot be easily shared with other elements on the page. There is a way to use CSS to format elements that are used frequently on a page. The CSS style definition for a page is located within the HEADER element. The following code shows where the CSS style is placed.

```
<head>
<meta content="text/html; charset=utf-8" http-
equiv="Content-Type" />
```

**This is Heading 1**

**Figure 2.5** The `style` attribute is used to format the H1 element.



```
<title>A sample CSS page</title>
<style type="text/css">
H1 {
    font-family: Verdana, Geneva, Tahoma,
sans-serif;
    font-size: 24px;
    color: #FF3300;
    font-weight: bolder;
}
</style>
</head>
```

This sample HTML code has moved the style definition for the H1 element into the style document. You can expand the document to format additional elements on the page. For instance, moving the CSS style definition to the top of the web page allows all of the P (paragraph) elements to look the same. The following CSS document placed in the HEAD section of the page will format all of the content on the screen.

```
<head>
<meta content="text/html; charset=utf-8"
http-equiv="Content-Type" />
<title>A sample CSS page</title>
<style type="text/css">
H1 {
    font-family: Verdana, Geneva, Tahoma,
sans-serif;
    font-size: 24px;
    color: #FF3300;
    font-weight: bolder;
}
h2 {
    font-family: Verdana, Geneva, Tahoma,
sans-serif;
    font-size: medium;
    color: #FF0000;
}
p {
    font-family: "Gill Sans", "Gill Sans MT",
Calibri, "Trebuchet MS", sans-serif;
    font-size: small;
}
```

## This is Heading 1

This is a paragraph. It's just something I threw together. This sentence contains [a link](#) to W3C, which you have probably visited. This sentence contains a link to [someplace else](#). This is [an offsite link](#). In the beginning was the word. And the word wrapped. I'll be looking soon for opportunities to style this text. In the meantime it will do merely to occupy space.

## I'm Heading 2

This is a paragraph. It's just something I threw together. In the beginning was the word. And the word wrapped. I'll be looking soon for opportunities to style this text. In the meantime it will do merely to occupy space.

**Figure 2.6** Placing the CSS style information within the HEAD element of a page allows elements to share the same design layout.

```
body {  
    margin: 2px}  
</style>  
</head>
```

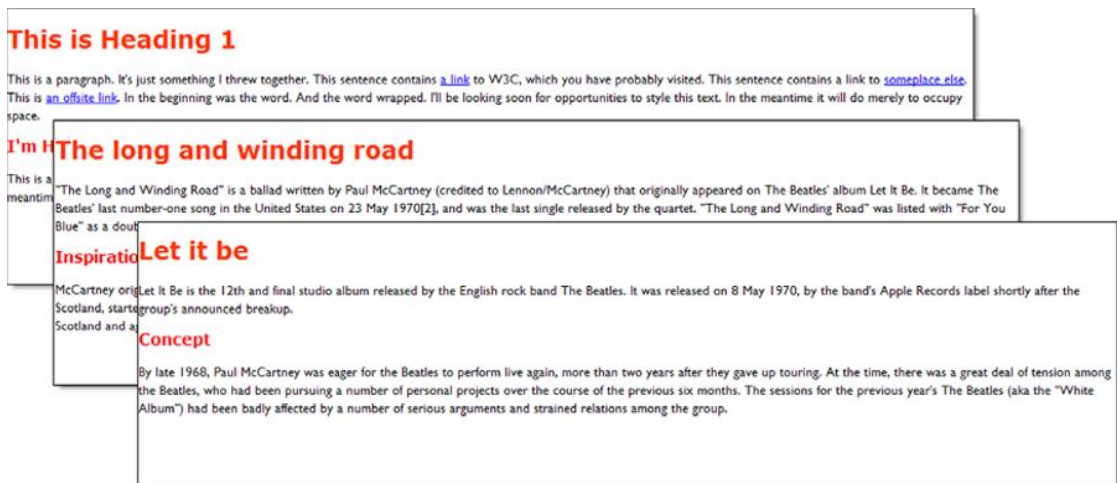
Figure 2.6 shows that the two paragraphs now look the same.

Cascading styles, however, have one additional trick up their sleeve. No web site is comprised of just one page. You have many, possibly hundreds or thousands, of pages in your web site. You do not want the burden of having to open each page and change the formatting each time you need to change the styles for your site.

Using CSS, you can now create a separate document in your site containing your style information and share it with all of your web pages. To create a shared CSS style document you need to create a text file with Notepad on a PC or TextEdit on a Mac. Copy your styles to the text file and save it to your site, naming the document with the extension .css. The final step is adding a line of code to the HTML that links to your web pages and points to the CSS document. The link is accomplished using the LINK element within the HEAD element in your web page. The following example is linking to a CSS document called "style.css."

```
<head>  
<meta content="text/html; charset=utf-8"  
http-equiv="Content-Type" />  
<title>A sample CSS page</title>  
<link href="style.css" rel="stylesheet"  
type="text/css" />  
</head>
```

The result is that you can create multiple web pages that share the same look and feel, as shown in Figure 2.7.



**Figure 2.7** Sharing a single CSS file allows the style formatting to be easily controlled over multiple web pages.

CSS is a very flexible design tool you can use to control the presentation of your content in any web page.

## The Format of CSS

Cascading Style Sheets is essentially a document that lists the visual presentation of your content. You have seen that there are different places you can store CSS information. There are different ways in which the CSS style definition can be applied to elements.

## Where to Get the Latest Information on CSS

The World Wide Web Consortium is the best place to go for the latest information on CSS. Check out the CSS Current Work Status page at <http://www.w3.org/Style/CSS/current-work>.

There are four main ways in which you can easily apply CSS to elements on a page:

- Modify an element's visual characteristics.
- Create a share class.
- Create a pseudo class.
- Create a pseudo element.

## Modifying Elements with CSS

Elements can be formatted directly in your code using the `style` attribute. More likely you will want to share the style you create with other elements on the page or site. Earlier in the

chapter an H1 and P element were modified with a custom style. To do this you should use the style sheet document.

You declare that you are going to modify an element in your style document by printing the element name. The following example demonstrates how to format a P element.

```
P {  
}
```

The curly brackets following the P element identify where you can place the formatting elements for the P tag. The following example shows style information for the P element.

```
p {  
  font-family: Arial, sans-serif;  
  font-size: medium;  
  color: #888;  
  padding-left: 25px;  
  padding-right: 50%;  
}
```

[Figure 2.8](#) illustrates how all of the P elements are modified by one style.

Styles can be shared among several elements if you want them to have a common style. For instance, the following example lists five different elements that each have different styles but are all the same color and font family.

```
<H1>Lorem Ipsum Header</H1>  
  
<p>In vestibulum, ipsum consectetur cursus  
porttitor, mi tellus euismod purus, ac egestas  
nisl risus ac risus. Suspendisse a nisi mi,
```

## Lorem Ipsum Header

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam accumsan convallis odio, vitae semper mi pretium laoreet.

In vestibulum, ipsum consectetur cursus porttitor, mi tellus euismod purus, ac egestas nisl risus ac risus. Suspendisse a nisi mi, nec rutrum nisi. Suspendisse pretium aliquet convallis.

Aliquam sollicitudin elementum est, commodo gravida lorem imperdiet ac. Donec rhoncus turpis vitae risus commodo ac mollis ligula aliquam. Donec in mi arcu, id vulputate turpis. Nullam nunc dui, euismod vel lobortis nec, suscipit non velit. Aliquam ornare, nibh eget facilisis lobortis, ligula velit suscipit sem, id condimentum est turpis ut magna. Morbi vitae hendrerit nibh.

In hac habitasse platea dictumst. Suspendisse eleifend ligula quis massa porta rutrum. Praesent in dolor laoreet leo interdum pulvinar sit amet quis lectus.

Proin lobortis, justo et rhoncus porta, neque orci pellentesque enim, quis sollicitudin nisl lacus eu nisi.

**Figure 2.8** A single P element style is shared by all P elements on the page.

nec rutrum nisi. Suspendisse pretium aliquet convallis. </p>

<h2>Lorem ipsum dolor sit amet</h2>

<p>Aliquam sollicitudin elementum est, commodo gravida lorem imperdiet ac. Donec rhoncus turpis vitae risus commodo ac mollis ligula aliquam. Donec in mi arcu, id vulputate turpis. Nullam nunc dui, euismod vel lobortis nec, suscipit non velit. Aliquam ornare, nibh eget facilisis lobortis, ligula velit suscipit sem, id condimentum est turpis ut magna. Morbi vitae hendrerit nibh. </p>

<h3>consectetur adipiscing elit</h3>

<p>In hac habitasse platea dictumst. Suspendisse eleifend ligula quis massa porta rutrum. Praesent in dolor laoreet leo interdum pulvinar sit amet quis lectus.</p>

<h4>Etiam accumsan convallis odio<h4>

<p>vitae semper mi pretium laoreet. </p>

The CSS is built up by applying first the shared styles between all five elements (H1, H2, H3, H4, and P). The first line in the style document lists all of the elements and then, between the curly brackets, the common font and color style is defined.

```
h1, h2, h3, h4, p{
  font-family: "Arial Narrow Bold", sans-serif;
  color: #CC3300;
}
```

Each element can now have its own style defined, as follows.

```
h1 {
  font-size: xx-large;
  font-weight: bolder;
}
h2 {
  font-size: medium;
}
h3 {
  font-size: small;
}
h4 {
  font-size: xx-small;
}
```

## Lorem Ipsum Header

In vestibulum, ipsum consectetur cursus porttitor, mi tellus euismod purus, ac egestas nisi risus ac risus. Suspendisse a nisi mi, nec rutrum nisi. Suspendisse pretium aliquet convallis.

Lorem ipsum dolor sit amet

Aliquam sollicitudin elementum est, commodo gravida lorem imperdiet ac. Donec rhoncus turpis vitae risus commodo ac mollis ligula aliquam. Donec in mi arcu, id vulputate turpis. Nullam nunc dui, euismod vel lobortis nec, suscipit non velit. Aliquam ornare, nibh eget facilisis lobortis, ligula velit suscipit sem, id condimentum est turpis ut magna. Morbi vitae hendrerit nibh.

consectetur adipiscing elit

In hac habitasse platea dictumst. Suspendisse eleifend ligula quis massa porta rutrum. Praesent in dolor laoreet leo interdum pulvinar sit amet quis lectus.

Etiam accumsan convallis odio

vitae semper mi pretium laoreet.

Proin lobortis, justo et rhoncus porta, neque orci pellentesque enim, quis sollicitudin nisi lacus eu nisi.

**Figure 2.9** Common style definitions can be applied to many different elements at once.

```
p {
    font-size: large;
    padding-left: 25px;
}
```

**Figure 2.9** shows how the common styles can be shared among the different elements.

The good news is that you can apply CSS to any element on the screen, including new HTML5 elements such as ASIDE, HEADER, FOOTER, SECTION, ARTICLE, and DIALOG. The following HTML can be formatted with CSS.

```
<H1>Sample Header</H1>
<ASIDE>
<H1>The Headline is formatted with CSS</H1>
<P>The PARAGRAPH element inherits the font style
formatting from the ASIDE element.</P>
<P>A link to another web page is added <a
href="http://www.focalpress.com">here</a>.</P>
</ASIDE>
```

You can use the following style to format the presentation of the ASIDE element.

```
aside {
    margin: 2px;
    border-style: dashed;
    font-family: Verdana, Helvetica, Arial,
sans-serif;
    font-size: 18px;
    line-height: 1.2em;
    text-align: left;
    position: absolute;
    color: #999;
```

```

background-color: ivory;
position: absolute;
left: 25px;
top: 75px;
width: 500px;
height: 250px;
}
a {
text-decoration: none;
color: #0000FF;
}
h1 {
font-size: 20px;
}
p{
font-size: 12px;
}

```

Figure 2.10 shows the results.



**Figure 2.10** All HTML5 elements, including new elements like ASIDE, can be styled with CSS.

## Creating Class Styles

There are times when you do not want all of the elements on the page to look the same. In fact, there are a lot of times when you want to apply custom styles to sections of text or to whole sections. The CSS class definition is your assistant in these situations.

The CSS class works in a very similar way to the elements' style definition. You define the CSS class either in the style region within your HEAD element or in the CSS style document. The following is an example of a CSS class style.

```
.mainTitleStyle {  
    font-family: Cambria, Cochin, Georgia, Times,  
    "Times New Roman", serif;  
    font-size: 30px;  
    font-weight: bolder;  
    color: #008000;  
}
```

As you can see, the main structure for defining the class style is the same as an element. The difference is that the class is identified by a leading period and the class name is all one word. You cannot use spaces in your class name.

After you have created your style you can apply it to any element in your web page. The element attribute `class` is used to associate the element with the CSS class. Here is an example.

```
<p class="mainTitleStyle">Lorem Ipsum Header</p>  
<p>In vestibulum, ipsum consectetur cursus  
porttitor, mi tellus euismod purus, ac egestas  
nisl risus ac risus. </p>  
<p class="mainTitleStyle">Lorem ipsum dolor sit  
amet</p>  
<p>Aliquam sollicitudin elementum est, commodo  
gravida lorem imperdiet ac. </p>  
<p class="mainTitleStyle">consectetur adipiscing  
elit</p>  
<p >Lorem ipsum dolor sit amet</p>
```

You can see that the P element is used for each line of text. The titles for each section are highlighted using the `class` attribute. [Figure 2.11](#) shows how the style looks in a web browser.

## Lorem Ipsum Header

In vestibulum, ipsum consectetur cursus porttitor, mi tellus euismod purus, ac egestas nisl risus ac risus.

## Lorem ipsum dolor sit amet

Aliquam sollicitudin elementum est, commodo gravida lorem imperdiet ac.

## consectetur adipiscing elit

Lorem ipsum dolor sit amet

**Figure 2.11** A custom CSS class is used to define the titles for each section.



There is no limit to the number of CSS class style definitions you can have. Class styles are very flexible and are used in many web sites. Check out the CSS styles for sites such as [www.bbc.co.uk](http://www.bbc.co.uk), [www.cnn.com](http://www.cnn.com), and [www.Google.com](http://www.Google.com) for examples of the CSS class used to define sections of HTML.

## Using Pseudo Class Styles

CSS gives you a third method for styling your content called a pseudo class, a special extension to the element style definition. The most common use for pseudo classes is with the ANCHOR element. The way an ANCHOR element, which identifies links on a web page, is defined in CSS is as follows.

```
a {
    text-decoration: none;
    color: #0000FF;
}
```

The ANCHOR element, however, completes several different activities. It has the default style, a different style when the link is being selected, a style for when the link has been visited, and a style for when you move your cursor over the link. Each of these different activities can be identified with pseudo classes. The following shows the pseudo class for a link that has been visited.

```
a:visited {
    color: #FF0000;
}
```

The ANCHOR element is listed first in your style document and is followed by a colon with the special pseudo class name called `visited`. In your web page, the visited link will now have a different color, as shown in [Figure 2.12](#).

The ANCHOR element has four pseudo classes: `link`, `active`, `hover`, and `visited`. The following style shows how you can define these four pseudo classes.

```
a{
    color: #0000FF;
}
a:link {
    text-decoration: none;
}
a:hover {
    text-decoration: underline;
}
a:active {
    text-decoration: line-through;
```

A link to a web page is added [here](#).

A link to another web page is added [here](#).

**Figure 2.12** Pseudo classes can be used to control different states of the ANCHOR element.

```

}
a:visited {
color: #FF0000;
}

```

The result is that you can now control the different actions of the ANCHOR tag.

CSS3 introduces additional pseudo class styles you can use. The complete list is:

- Active—the active element
- Focus—the element with focus
- Visited—a visited link
- Hover—the state when your cursor is over a link
- Link—an unvisited link
- Disabled—the state of an element when it has been disabled
- Enabled—the state of an element when it has been enabled
- Checked—a form element that has been checked
- Selection—when a user selects a range of content on the page
- Lang—the designer can choose which language is used for the style
- Nth-child(n)—an element that is a specified child of the first sibling
- Nth-last-child(n)—an element that is a specified child of the last sibling
- First-child—the first use of an element on the page
- Last-child—the last use of an element on the page
- Only-child—the only use of a element on the page

## Using Pseudo Elements

New to CSS3 is a new extension called pseudo elements. A pseudo element allows you to control aspects of an element in the page. For instance, you may want special text treatment for the first letter of each paragraph you write. There are four pseudo elements you can use:

- First-letter
- First-line
- Before
- After

The definition for pseudo elements is very similar to pseudo classes. The following style applies first-letter pseudo element styles to the P element. Note that the pseudo element leads with two colons.

```

p::first-letter {
font-size: 60px;
}

```

**A** link to a web page is added [here](#).

**A** link to another web page is added [here](#).

**Figure 2.13** Pseudo elements modify specific parts of the element.

**Figure 2.13** illustrates how this is presented in the web browser.

Note how the leading letter of each line is much larger than the rest of the line. At this time there are few pseudo elements.

## Designing Your Web Page with CSS

CSS is much easier to master than more complex parts of HTML5 such as Web Workers, Geo Location, and JavaScript. The basic premise for all CSS is that you have a definition that requires a value. For instance, if you want to define the size of a particular font, you write the correct CSS definition (font-size) and place a value; for example:

```
font-size: 60px;
```

There are four rules you must follow:

1. Use a valid CSS definition.
2. Place a colon after the definition.
3. Add a valid value for the definition.
4. Complete the statement with a semi-colon.

Follow these four rules and you are golden.

## Tools to Help with Your CSS Designs

For basic CSS manipulation there are some great tools you can use. Adobe's Dreamweaver and Microsoft's Expression Web both support CSS2 design definition. Both of these tools offer visual editors you can easily use to write CSS. Unfortunately, your choices drop significantly when you start looking for more advanced CSS3 tools. This is in part due to the rapid development of CSS3. Check out [www.visualizingtheweb.com](http://www.visualizingtheweb.com) for the latest information on CSS3 tools.

When CSS was first released in 1997 there were about a dozen or so definitions to control visual aspects such as font size, color, and background color. Now there are hundreds of different definitions that can be used extensively with any element on the screen.

## Controlling Font Display with CSS

One of the easiest places to start learning how to use CSS definitions is through font control. CSS1 and CSS2 support nine different definitions within the font-family:

- Font-family
- Font-size
- Color
- Text-shadow
- Font-weight

- Font-style
- Font-variant
- Text-transform
- Text-decoration

The font-family definition allows you to select a font for your design. Here is how to write the definition:

```
font-family: Arial;
```

The challenge in using the font-family definition is that the number of fonts you can select from is limited to the fonts installed on the computer of the person who is viewing your web page. Web browsers and operating systems install a core set of fonts that you can use in your designs. The list of fonts available that are “Web safe” include:

- Arial/Helvetica
- Times New Roman/Times
- Courier New/Courier
- Verdana
- Georgia
- Comic Sans MS
- Trebuchet MS
- Arial Black
- Impact
- Palatino
- Garamond
- Bookman
- Avant Garde

This list is not very exhaustive and you run into issues where the fonts will not match. For instance, you may select the font Tahoma and it will look great on Windows XP, Vista, and Windows 7, but will not look the same on a Mac or iPhone. Often you will find that there are similar fonts on Windows and Mac computers, but they simply have different names. For instance, you can select the following font-family:

```
font-family: "Courier New", Courier, monospace;
```

This collection of fonts will allow the text to be presented correctly no matter the system viewing the page. In this instance, “Courier New” is the PC name for “Courier” on the Apple Mac; “monospace” is a Unix/Linux equivalent.

Here is a collection of safe font-family names you can use:

- Arial, Arial, Helvetica, sans serif
- Arial Black, Arial Black, Gadget, sans serif
- Comic Sans MS, Comic Sans MS, cursive
- Courier New, Courier New, Courier, monospace
- Georgia, Georgia, serif
- Impact, Impact, Charcoal, sans serif

- Lucida Console, Monaco, monospace
- Lucida Sans Unicode, Lucida Grande, sans serif
- Palatino Linotype, Book Antiqua, Palatino, serif
- Tahoma, Geneva, sans serif
- Times New Roman, Times, serif
- Trebuchet MS, Helvetica, sans serif
- Verdana, Verdana, Geneva, sans serif
- Wingdings, Zapf Dingbats (Wingdings, Zapf Dingbats)

## Embedding Fonts Using CSS3

A way to get around the problems of creating font-family lists is to embed the font directly into the CSS. CSS3 finally allows you to do this across your web browsers. The technology for font embedding, however, is not new. Netscape Navigator 4 was the first web browser that allowed you to support font embedding using a plug-in technology called TrueDoc by Bitstream. To compete with Navigator 4, Microsoft released a “me too” technology called Embedded Open Type (EOT) in the Windows version of Internet Explorer 4. The technology has not been removed from the Microsoft browser and is still supported in Internet Explorer 8.

Embedded Open Type is a method of creating a file that can be downloaded to the web browser. The file is an EOT file. To protect the copyright of the original font developer the EOT file is created using a font outline of the original font. You can download the free Microsoft Web Embedding Font Tool (WEFT) from Microsoft to create your own EOT files (<http://www.microsoft.com/typography/webembedding/>).

The EOT format is not an open format and has not been adopted by modern web browsers or embraced by W3C.

Without a shared standard for embedding fonts, designers have been forced to use other techniques to emulate font embedding. These have included creating JPEG images of text with custom fonts or using third-party plug-in technologies such as Adobe’s Flash.

As you might expect, HTML5 has driven new technologies to enable true font embedding. Three standards are now recommended to embed fonts:

- TrueType
- OpenType
- Scalable vector graphic fonts

It is quite likely that you already have TrueType and OpenType fonts installed on your computer. They are, by default, the standard Windows font format. SVG fonts are more complex and will be covered in more detail in *Rendering HTML5 Illustration*.

Embedding a font into your CSS document is now very easy. [Figure 2.14](#) shows text in a web page using a custom font.

In hac habitasse platea dictumst.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam accumsan convallis odio, vitae semper mi pretium laoreet.

In vestibulum, ipsum consectetur cursus porttitor, mi tellus euismod purus, ac egestas nisi risus ac risus. Suspendisse a nisi mi, nec rutrum nisi. Suspendisse pretium aliquet convallis.

Aliquam idlichudin elementum est, commodo gravida lorem imperdiet ac.

In hac habitasse platea dictumst.

Donec rhoncus turpis vitae risus commodo ac mollis ligula aliquam.

Donec in mi arcu, id vulputate turpis.

Nullam nunc dui, euismod vel lobortis nec, suscipit non velit.

Aliquam ornare, nibh eget facilisis lobortis, ligula velit suscipit sem, id condimentum est turpis ut magna.

**Figure 2.14** CSS3 now allows you to embed TrueType and OpenType fonts directly into your web pages.

*This text is an  
embedded font*

**Figure 2.15** The TrueType font BlackJar.ttf can now be embedded into a web page.

To embed a font into a web page you need only two things: the font file and the definition in CSS linking to the font. The font BlackJar.ttf is used in [Figure 2.14](#). [Figure 2.15](#) shows you what the TrueType font looks like.

You need to create a new font-family in your CSS document that links to the TrueType font. The following CSS code shows, in line 2, that you are creating a new font-family called “BlackJar” and, in line 3, you are linking to the font and identifying the type of font.

```
@font-face{
font-family: 'BlackJar';
src: url('BLACKJAR.ttf') format('truetype');
}
```

You now have a new font-family that you can reference in your normal CSS. Here, a P element is being formatted using the new font-family.

```
p {
text-align: center;
font-family: 'BlackJar';
font-size: 3cm;
}
```

You can now use the font within your page design. If you want to also use the font with Internet Explorer you can add the Embedded Open Type with your new font-family. You only need to modify the @font-face description as follows.

```
@font-face{
font-family: 'BlackJar';
```

```
src: url('BLACKJAR.ttf') format('truetype');  
src: url('BLACKJAR.eot');  
}
```

The fourth line links to an EOT version of the BlackJar font. You will notice that you do not need to add a format value for EOT fonts. Now your web pages will display correctly no matter what web browser is viewing your design. Font freedom has finally come to the Web!

## Sizing Fonts with CSS Units of Measurement

After selecting a font-family for your text you will also want to select the size of the font. By default, all web browsers have a preinstalled definition for a standard font size. This font size is usually 12 point (pt). You can use this as a size for your fonts as they appear on the screen using the following CSS font-size definition:

```
Font-size:medium;
```

If you want your font to appear smaller or larger on the screen you can use the following sizes for your fonts:

- Xx-small (approximately 7.5 pt)
- X-small (approximately 9 pt)
- Small (approximately 10 pt)
- Medium (approximately 12 pt)
- Large (approximately 14 pt)
- X-large (approximately 18 pt)
- Xx-large (approximately 24 pt)
- Smaller
- Larger

Each of these font sizes are relative to the core browser-defaulted font size. If the person who owns the web browser has changed that default, then the sizes will be dynamically changed.

As a designer you are limited by the default font-size list. The good news is that CSS allows you to leverage units of measurement to add precise size to your font. The following are all valid CSS units of measurement you can use:

- cm—centimeter
- in.—inch
- mm—millimeter
- pc—pica (1 pc = 12 pts)
- pt—point (1 pt =  $\frac{1}{72}$  in.)
- px—pixels
- rem—font size of the root element

Using these different font sizes, the following styles are all valid.

```
.default {
    font-family: "Segoe UI", Tahoma, Geneva,
Verdana;
    font-size: medium;
}
.px {
    font-family: "Segoe UI", Tahoma, Geneva,
Verdana;
    font-size: 15px;
}
.cm {
    font-family: "Segoe UI", Tahoma, Geneva,
Verdana;
    font-size: .5cm;
}
.mm {
    font-family: "Segoe UI", Tahoma, Geneva,
Verdana;
    font-size: 2mm;
}
.inch {
    font-family: "Segoe UI", Tahoma, Geneva,
Verdana;
    font-size: .25in;
}
.pica {
    font-family: "Segoe UI", Tahoma, Geneva,
Verdana;
    font-size: 2pc;
}
.point {
    font-family: "Segoe UI", Tahoma, Geneva,
Verdana;
    font-size: 10pt;
}
.rem {
    font-family: "Segoe UI", Tahoma, Geneva,
Verdana;
    font-size: 1rem;
}
```

**These font styles are applied to the following HTML code.**

```
<p class="default">In hac habitasse platea
dictumst. </p>

<p class="px">Lorem ipsum dolor sit amet,
consectetur adipiscing elit. Etiam accumsan
convallis odio, vitae semper mi pretium laoreet.
</p>
```



`<p class="cm">In vestibulum, ipsum consectetur  
cursus porttitor, mi tellus euismod purus, ac  
egestas nisl risus ac risus. Suspendisse a nisi  
mi, nec rutrum nisi. Suspendisse pretium aliquet  
convallis. </p>`

`<p class="mm">Aliquam sollicitudin elementum est,  
commodo gravida lorem imperdiet ac. </p>`

`<p class="inch">In hac habitasse platea dictumst.  
</p>`

`<p class="pica">Donec rhoncus turpis vitae risus  
commodo ac mollis ligula aliquam. Donec in mi  
arcu, id vulputate turpis. </p>`

`<p class="point">Nullam nunc dui, euismod vel  
lobortis nec, suscipit non velit. </p>`

`<p class="rem">Aliquam ornare, nibh eget  
facilisis lobortis, ligula velit suscipit sem, id  
condimentum est turpis ut magna. </p>`

Figure 2.16 shows how these fonts are presented in your web browser.

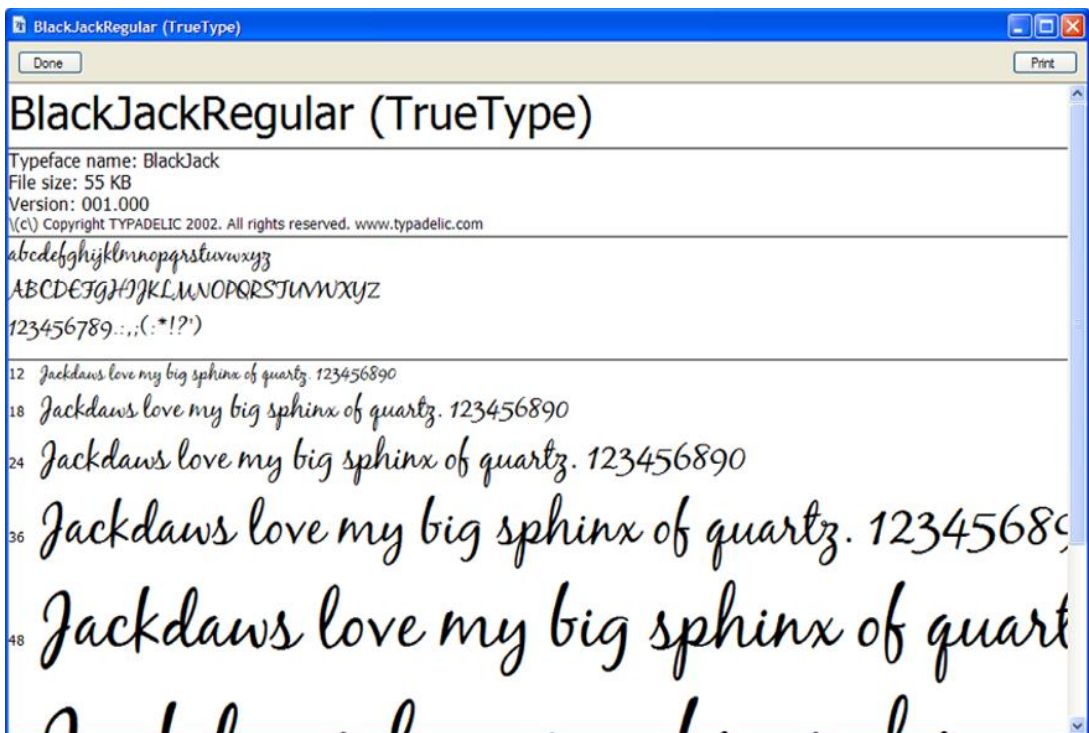


Figure 2.16 You have absolute control over your text using the many different units of measure available in CSS3.

## Color Control for Fonts

As with size, color has many different units of measure. The default for Web design is hexadecimal, a combination of six letters and numbers. CSS3 provides a much broader palette of colors to choose from that include:

- Color name—you can create names for colors such as brown, black, red, or even cyan
- Full hexadecimal—a hexadecimal value comprised of six alpha-numeric values
- Short hexadecimal—a hexadecimal value comprised of three alpha-numeric values
- RGB—a combination of red, green, and blue values
- RGBA—a combination of red, green, and blue values with a transparency value (alpha)
- HSL—a combination of hue, saturation, and lightness
- HSLA—a combination of hue, saturation, and lightness with a transparency value (alpha)

The following CSS uses these values to show you can create the color red in several different ways.

```
.name {  
color: red;  
}  
.fullHexVersion {  
color: #FF0000;  
}  
.shortHexVersion {  
color: #F00;  
}  
.rgb {  
color: rgb(255,0,0);  
}  
.rgba {  
color: rgba(255,0,0,100);  
}  
.hsl {  
color: hsl(0%, 100%, 50%);  
}  
.hsla {  
color: hsl(0%, 100%, 50%, 100%);  
}
```

These different values are used in different places within the design community.

## Adding Drop Shadow Text Effects

Love them or hate them, you cannot get away from the handy design technique of drop shadows. CSS3 now supports drop shadow effects and they are very easy to add to your designs.

There are four elements that you can use to control the drop shadow definition:

- horizontal-offset (length, required)
- vertical-offset (length, required)
- blur-radius (length, optional)
- shadow-color (color, optional)

The following CSS definition is an example of the use of drop shadow.

```
.dropShadow {
    font-family: "Segoe UI", Tahoma, Geneva,
Verdana;
    font-size: medium;
    color: #CC3300;
    text-shadow: 0.25em 0.25em 2px #999;
}
```

The effect draws a light-gray drop shadow with a slight blur, as shown in [Figure 2.17](#).

Different colors and units of measurement can be used with the drop shadow effect. The following CSS definition uses pixels and RGBA for the measurement and color.

```
.transparentDropShadow {
    font-family: "Segoe UI", Tahoma, Geneva, Verdana;
    font-size: 15px;
    color: rgba(255,0,0,1);
    text-shadow: 5px 5px 5px rgba(0, 0, 0, 0.5);
}
```

Finally, you can use the drop shadow effect to force a “cut-out” effect with your text. Apply the following CSS to text on the screen.

```
.cutout {
    font-family: "Segoe UI", Tahoma, Geneva,
Verdana;
    font-size: 2pc;
    color: white;
    text-shadow: 0em 0em 2em black;
}
```

*In hac habitasse platea dictumst.*

**Figure 2.17** CSS now allows you to add drop shadows to your text.

In hac habitasse platea dictumst.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam accumsan convallis odio, vitae semper mi pretium laoreet.

Donec rhoncus turpis vitae risus commodo ac mollis ligula aliquam.

Donec in mi arcu, id vulputate turpis.



**Figure 2.18** You can use the drop shadow to create cut-out effects.

Figure 2.18 demonstrates the effect of the drop shadow as a cut out.

## Additional Font Definitions

The remaining values within the font-family do not have ranges and can be summarized as:

- Font-weight—boldness value from 100–900, with 300 being normal
- Font-style—italic, normal, oblique
- Font-variant—normal, small caps
- Text-transform—capitalize, lowercase, normal
- Text-decoration—underline, overline, line-through, none, blinking

As you can see, CSS gives you an amount of control over how your text is displayed on the screen.

## Working with Columns in CSS3

A challenge for any web page is to create content that is split over two or more columns on the page. Creating columns often requires using complex tables structured together. Though not strictly part of the text family of CSS definitions, the new multi-column layout is best at home when used with text on the screen.

The goal of the multicolumn definition is to allow your content to be spread evenly over two or more columns. There are three parts to a column layout:

- Number of columns
- Gap between the columns
- Column design (optional)

The following CSS demonstrates how you can set up multiple columns to display in Safari/Chrome and FireFox.

```
.simple {
    font-family: "Segoe UI", Tahoma, Geneva,
    Verdana;
    font-size: 12px;
    color: #444;
    text-align: justify;
    -moz-column-count: 4;
    -moz-column-gap: 1em;
    -webkit-column-count: 4;
    -webkit-column-gap: 1em;
}
```

In this example, the column count is four and the gap is 1em. [Figure 2.19](#) shows how this is displayed in your web browser.

You can add a column design between each column. The structure is as follows.

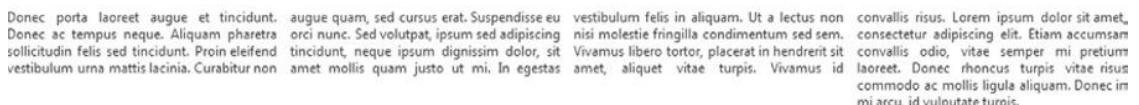
```
-moz-column-rule: 1px solid #222;
-webkit-column-rule: 1px solid #222;
```

For each column design you can identify the width, border style, and color. You can use the standard measurement and color CSS formatting. The number of border styles you have to choose from is:

- None
- Hidden
- Dotted
- Dashed
- Solid
- Double
- Groove
- Ridge
- Inset
- Outset

Additional elements, such as the IMG, can be used with text content in the column layout. [Figure 2.20](#) illustrates a complex use of a multicolumn layout.

Below is an example of a multi-column layout



Donec porta laoreet augue et tincidunt. Donec ac tempus neque. Aliquam pharetra sollicitudin felis sed tincidunt. Proin eleifend vestibulum urna mattis lacinia. Curabitur non augue quam, sed cursus erat. Suspendisse eu orci nunc. Sed volutpat, ipsum sed adipiscing tincidunt, neque ipsum dignissim dolor, sit amet mollis quam justo ut mi. In egestas vestibulum felis in aliquam. Ut a lectus non nisi molestie fringilla condimentum sed sem. Vivamus libero tortor, placerat in hendrerit sit amet, aliquet vitae turpis. Vivamus id convallis risus. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam accumsan convallis odio, vitae semper mi pretium laoreet. Donec rhoncus turpis vitae risus commodo ac mollis ligula aliquam. Donec in mi arcu, id vulputate turpis.

**Figure 2.19** A simple, four-column layout.

Below is an example of a multi-column layout

Donec porta laoreet augue et tincidunt. Donec ac tempus neque. Aliquam pharetra sollicitudin felis sed tincidunt. Proin eleifend vestibulum urna mattis lacinia. Curabitur non augue quam, sed cursus erat. Suspendisse eu orci nunc. Sed volutpat, ipsum sed adipiscing tincidunt, neque ipsum dignissim dolor, sit amet mollis quam justo ut mi. In egestas vestibulum felis in aliquam. Ut a lectus non nisi molestie fringilla condimentum sed sem. Vivamus libero tortor, placerat in hendrerit sit amet, aliquet vitae turpis. Vivamus id convallis risus. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam accumsan convallis odio, vitae semper mi pretium laoreet. Donec rhoncus turpis vitae risus commodo ac mollis ligula aliquam. Donec in mi arcu, id vulputate turpis.



Donec ac tempus neque. Aliquam pharetra sollicitudin felis sed tincidunt. Proin eleifend vestibulum urna mattis lacinia.

Curabitur non augue quam, sed cursus erat. Suspendisse eu orci nunc. Sed volutpat, ipsum sed adipiscing tincidunt, neque ipsum dignissim dolor, sit amet mollis quam justo ut mi. In egestas vestibulum felis in aliquam. Ut a lectus non nisi molestie fringilla condimentum sed sem. Vivamus libero tortor, placerat in

hendrerit sit amet, aliquet vitae turpis. Vivamus id convallis risus. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam accumsan convallis odio, vitae semper mi pretium laoreet. Donec rhoncus turpis vitae risus commodo ac mollis ligula aliquam. Donec in mi arcu, id vulputate turpis.

**Figure 2.20** Columns can have decoration between each column.

The CSS to create this layout is as follows.

```
.complex {
    font-family: "Segoe UI", Tahoma, Geneva,
    Verdana;
    font-size: 1.2pc;
    color: #444;
    text-align: left;
    -moz-column-count: 3;
    -moz-column-gap: 1em;
    -moz-column-rule: 2px dotted #999;
    -webkit-column-count: 3;
    -webkit-column-gap: 1em;
    -webkit-column-rule: 2px dotted #999;
}
```

The style in this column layout is applied to a P element that contains both text and an IMG element. You should experiment with columns—they are certainly much easier to use than complex tables.

## Using CSS3 to Control Visual Display

While most of your time working with CSS will be formatting text, CSS is not just the domain of text on the screen. Indeed, CSS is expanding in scope to allow you to control as much as possible on the screen. The final two sections look at CSS control over static and animated elements on the screen.

## Positioning Design Elements with CSS

Have you used HTML tables to align elements in your web page? If so, it is a pain in the neck, isn't it? Using tables to control layout presents two key problems. First, the layout is flat with

images “sliced” to fit correctly on the screen. Second, if you want to make a change to your design you need to relayout the whole table. Tables are simply not a sensible solution for design layout.

Fortunately, all web browsers now support positioning within CSS to give you absolute control over your design. The difference between using positioning in CSS to using HTML tables is dramatic. First, an element can be placed at any point on the screen using CSS positioning. There are no limitations. A second benefit is that you can layer elements on top of each other. If you are used to working with layers in PhotoShop or a similar graphics tool then you already understand the value of this feature. Layers allow you to segment regions of your design. When you need to make changes to your design, you only need to modify the layer with your content.

Use the following definitions to position an element on the screen:

- Position—values include absolute, relative, fixed, inherit
- Width—the width of the layer
- Height—the height of the layer
- Left—where from the left margin the layer starts
- Top—where from the top margin the layer starts
- Overflow—how to present content that goes beyond the scope of the layer
- Z-index—the stack order for layers on the screen

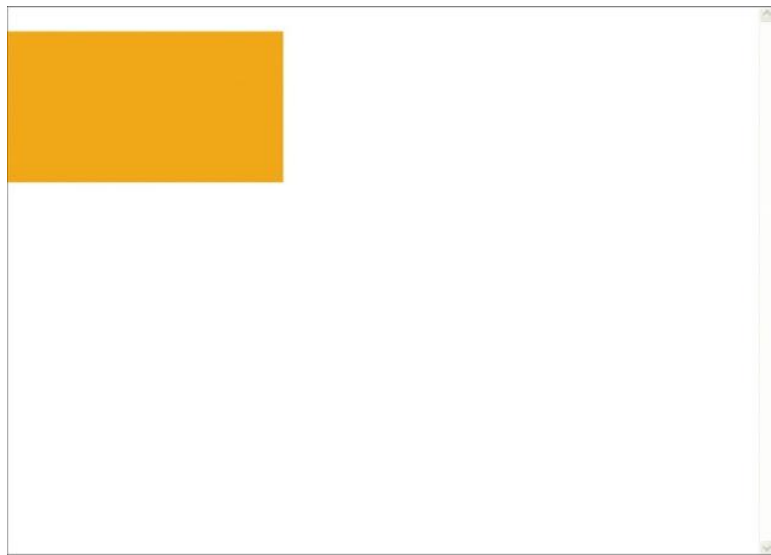
The following CSS can be applied to any HTML element to control the positioning of the element on the screen.

```
.firstLayer{
  Background-color: orange;
  position: absolute;
  width: 295px;
  height: 160px;
  z-index: 1;
  left: 439px;
  top: 28px;
  overflow
}
```

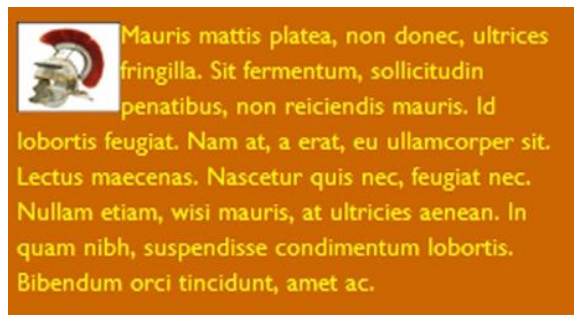
The following HTML code has the layer definition applied to it using a CSS class:

```
<article class="firstlayer"></article>
```

The orange box in [Figure 2.21](#) is okay by itself, but what can you do with it? The position of any HTML element can be controlled with positioning. The orange box in this example is an HTML ARTICLE element. Any text, tables, graphics, or other HTML elements can be inserted into the ARTICLE. [Figure 2.22](#) illustrates an image wrapped by text inside of an ARTICLE element positioned absolutely on the page.



**Figure 2.21** The positioning of the orange box is controlled using CSS positioning.



**Figure 2.22** Content within a layered element inherits the positioning of the parent element.



(a)



(b)

**Figure 2.23** You can create (a) linear and (b) radial gradients in CSS3.

Positioning is key to your design work in your web pages. With wide support of CSS positioning, you now have the control you need.

## Increase Control Over Color

Control over the use of color has increased significantly with CSS3. You saw earlier that you can now use long hexadecimal, short hexadecimal, RGB, RGBA, HSL, and HSLA to have access to millions of colors. In addition to solid colors, CSS3 gives you the ability to add gradients.

You can currently create two different types of gradient: linear and radial, as shown in [Figure 2.23](#).



The gradient definition is comprised of several key elements:

- Type—either radial or linear
- Point—two space-separated values that explain where the gradient starts (this can be achieved with a number, percentage, or using the keywords “top,” “bottom,” “left,” and “right”)
- Radius—the radius is a number that you only need to specify when you use the radial type
- Stop—the function of the `stop` value is to identify the blend strength as a percentage or number between 0 and 1 (such as 0.75 or 75%) and a color. You can use any CSS3-supported color.

Putting all of these together will give you a gradient. Gradients can be used with the following definitions:

- Background-image
- Border-image
- List-style-image
- Content property

The following example adds a gradient that goes from red to orange to yellow.

```
body {  
background-image: -webkit-gradient(linear, left  
top, left bottom, from(red), to(yellow), color-  
stop(0.5, orange), color-stop(0.5, orange));}
```

As you can see, the gradient is substituting an image in the background-image definition. The first definition identifies the gradient as linear. The next definition explains the gradient is going to go from top to bottom. The two elected colors are red and yellow. The stop function has the colors blending halfway through to orange. The result is displayed in [Figure 2.24](#).

A radial gradient is completed in a similar way. The following adds a radial gradient that moves from red to orange to yellow.



**Figure 2.24** Gradients can be used to create colored backgrounds.

```
body {
background-image: -webkit-gradient(radial, 45 45,
15, 100 100, 250, from(red), to(yellow),
color-stop(50%, orange));}
```

In this instance, the numbers following the radial declaration determine the shape of the radius. The first two numbers dictate the angle of the ellipse in degrees. The third number dictates the size of the inner circle. The fourth and fifth numbers dictate the position of the gradient (left and top). The final number dictates the final size of the radius. [Figure 2.25](#) is the result.

Currently, gradients are only supported in Safari and Chrome. This is expected to change with FireFox 4.0.



**Figure 2.25** Gradients can be used to create colored backgrounds.

## Multiple Background Objects

You quickly run into limitations when you can use only one background image. With CSS3 you can now run multiple background images. Any element that supports the background-image definition now supports multiple background images. Using background images is very easy. You can start by listing the images you want to use. Take for instance the following code.

```
background-image: url(http://upload.wikimedia.
org/wikipedia/commons/3/36/Team_Singapore_
fireworks_display_from_Singapore_Fireworks_
Festival_2006.jpg), url(http://upload.wikimedia.
org/wikipedia/commons/b/b2/OperaSydney-
Fuegos2006-342289398.jpg);
```

You can specify where you want each background to appear on the screen using the background-position definition. The definition is paired for the position of the background:

```
background-position: bottom left, top right;
```



**Figure 2.26** The two images are being used as background images

Figure 2.26 shows the end result.

As you might expect, you can mix gradients and multiple background images together. The following CSS blends a radial gradient with two background images.

```
<html>
<head>
<title>Multiple Backgrounds</title>
  <style>
    body {
      background-image:
        url(http://upload.wikimedia.org/wikipedia/
        commons/3/36/Team_Singapore_fireworks_display_
        from_Singapore_Fireworks_Festival_2006.jpg),
        url(http://upload.wikimedia.org/wikipedia/
        commons/b/b2/OperaSydney-Fuegos2006-342289398.
        jpg), -webkit-gradient(radial, 45 45, 15,
        100 100, 250, from(gold), to(magenta), color-
        stop(50%, black));
      background-repeat: no-repeat;
      background-position: bottom left, top right;
      background-color:black;}
    </style>
  </head>
<body>
</body>
</html>
```



**Figure 2.27** Images and gradients can be mixed to create unique background images.

Figure 2.27 shows the results.

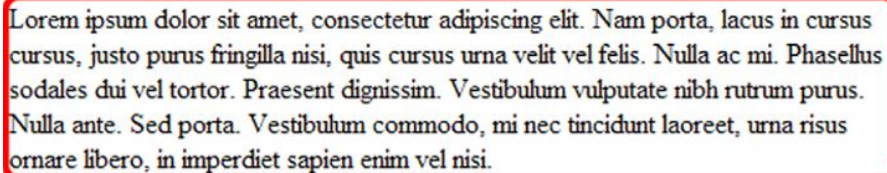
Again, as with gradients, multiple backgrounds are not currently supported by FireFox.

## Adding Rounded Corners to Layers

Adding rounded corners is not a new technique for the Web. Many web sites use this technique. The effect, however, is created through using images and tables to create the illusion of rounded corners. Adding images to the pages ensures that the page takes longer to load and makes modifying the page later more complex.

A simpler approach is to use the proposed corner-radius CSS definition that is currently supported in FireFox 3.0, Safari 3.0, Mobile Safari on your iPhone/iPod Touch, and Google's Chrome. The corner-radius definition is a line you can add to your CSS style. The following HTML code has a style embedded that changes the presentation of the block of text to have rounded corners with a heavy, black outline.

```
<p style="-moz-border-radius: 10px;-webkit-  
border-radius: 10px;border: 4px solid #FF0000;">  
Lorem ipsum dolor sit amet, consectetur  
adipiscing elit. Nam porta, lacus in cursus  
cursus, justo purus fringilla nisi, quis cursus  
urna velit vel felis. Nulla ac mi. Phasellus  
sodales dui vel tortor. Praesent dignissim.  
Vestibulum vulputate nibh rutrum purus. Nulla  
ante. Sed porta. Vestibulum commodo, mi nec  
tincidunt laoreet, urna risus ornare libero, in  
imperdiet sapien enim vel nisi.</p>
```



Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nam porta, lacus in cursus  
 cursus, justo purus fringilla nisi, quis cursus urna velit vel felis. Nulla ac mi. Phasellus  
 sodales dui vel tortor. Praesent dignissim. Vestibulum vulputate nibh rutrum purus.  
 Nulla ante. Sed porta. Vestibulum commodo, mi nec tincidunt laoreet, urna risus  
 ornare libero, in imperdiet sapien enim vel nisi.

**Figure 2.28** Layers can now have rounded corners.

The style description has been highlighted in bold. Your content will now look like [Figure 2.28](#) in your web page.

As you can see, the block of text now has a solid red line with rounded corners. It is this style description that controls the size of the radius, not an image. You can then easily modify the description as shown in the following.

```
-moz-border-radius: 10px
-webkit-border-radius: 10px
```

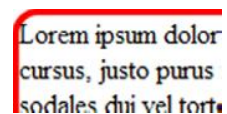
The standard is currently only in the proposal stage and has not been adopted by all web browsers. For this reason, you need to add two border-radius style descriptions: one for FireFox (-moz-border-radius), and one for Safari/Chrome (-webkit-border-radius). Changing the value of the border-radius will change the size of the border. For instance:

```
Border-radius: 15 px
Border-radius: 25 px
Border-radius: 45 px
```

[Figures 2.29 through 2.31](#) shows some border-radius examples.

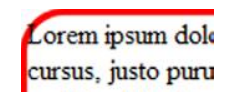
As you increase the border-radius, you will also have to add additional styles, such as padding, to ensure that your border does not cut through the text as is shown in the example of border-radius: 45 px. Here is how you can add padding to manage your style.

```
<p style="-moz-border-radius: 45px;-webkit-  
border-radius: 45px;border: 4px solid #FF0000;  
padding: 12px;">Lorem ipsum dolor sit amet,  
consectetur adipiscing elit. Nam porta, lacus  
in cursus cursus, justo purus fringilla nisi,  
quis cursus urna velit vel felis. Nulla ac  
mi. Phasellus sodales dui vel tortor. Praesent  
dignissim. Vestibulum vulputate nibh rutrum  
purus. Nulla ante. Sed porta. Vestibulum commodo,  
mi nec tincidunt laoreet, urna risus ornare  
libero, in imperdiet sapien enim vel nisi.</p>
```



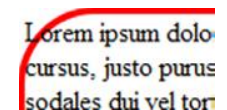
Lorem ipsum dolor  
 cursus, justo purus  
 sodales dui vel tortor

**Figure 2.29** The border-radius controls how round the corners are.



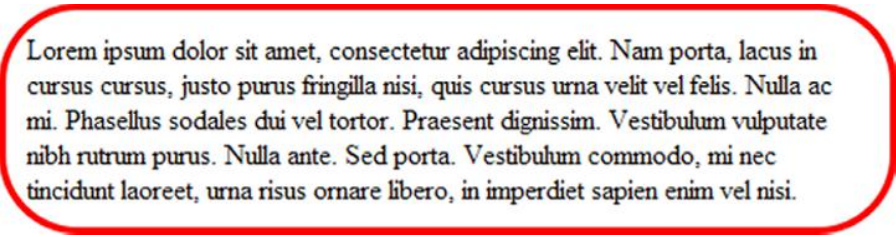
Lorem ipsum dolo  
 cursus, justo puru

**Figure 2.30** A border-radius of 25 pixels.



Lorem ipsum dolo  
 cursus, justo purus  
 sodales dui vel tort

**Figure 2.31** A border-radius of 45 pixels.



Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nam porta, lacus in cursus cursus, justo purus fringilla nisi, quis cursus urna velit vel felis. Nulla ac mi. Phasellus sodales dui vel tortor. Praesent dignissim. Vestibulum vulputate nibh rutrum purus. Nulla ante. Sed porta. Vestibulum commodo, mi nec tincidunt laoreet, urna risus ornare libero, in imperdiet sapien enim vel nisi.

**Figure 2.32** The layer now looks like a rectangle with rounded corners.

Figure 2.32 shows how the content looks.

The new border-radius style also has the option of allowing you to control which corner you want the border to appear on. This can be useful when you want to create tabs for your web page. For instance, the following style will add tabs to the top left and top right corners.

```
.standardTabEffect{
    font-family: Arial, Helvetica, sans-serif;
    font-size: 15px;
    background-color: #FFFF00;
    -moz-border-radius-topleft: 15px;
    -moz-border-radius-topright: 15px;
    -webkit-border-radius-topleft: 15px;
    -webkit-border-radius-topright: 15px;
    border: 4px solid #FF0000;
    padding: 10px;
    color: #FF0000;
    text-decoration: none;
    font-weight: bold;
}
```

This style can now be added to a central style sheet link to the content on your page. The content on your page can now reference the style. You can add the following HTML code to see this effect.

```
<a class="standardTabEffect" href="#">This is Tab 1</a><a class="standardTabEffect" href="#">This is Tab 2</a><a class="standardTabEffect" href="#">This is Tab 3</a>
```

Figure 2.33 shows how the HTML code will look when you view the page.



**Figure 2.33** The border-radius is used to create tabbed buttons.

As you might imagine, you can inherit existing CSS formatting into your border-radius designs. For instance, you can add a simple rollover effect when you include the following style description. The important part is to add the `:hover` parameter. This instructs the web browser to only use this style when a user is rolling over the link with the mouse.

```
.standardTabEffect:hover{
    background-color: #FF0000;
    border: 4px solid #FFFF00;
    color: #FFFF00;
}
```

Figure 2.34 shows what the action looks like.

Without using complex images or tables, you have created a series of tabs that can be easily managed through CSS and HTML.



**Figure 2.34** The hover pseudo class can add an effect as you move the mouse over a button.

## Dazzling Your Audience with CSS3 Animation

CSS3 continues to expand what you can visually accomplish in your web pages. Animation is now also available to you as a design choice. Animation is split into two key parts: transitions and transforms.

- Transitions control the change of state for an element, such as text fading in or changing color.
- Transforms control the placement of an element.

The following two sections explain how you can control these two new animation techniques in your CSS designs.

### Using Transitions in CSS

The transition effect is best used when you create a class and then a hover pseudo class to illustrate when the effect is to happen (i.e., when your cursor moves over the element). The transition itself is made of three parts:

- Property—the linked property between the two classes
- Duration—how long in seconds the transition will take
- Timing function

The timing function keywords have the following definitions:

- Linear—the linear function just returns as its output the input that it received.



- Ease—the default function, ease, is equivalent to cubic-bezier (0.25, 0.1, 0.25, 1.0).
- Ease-in—the ease-in function is equivalent to cubic-bezier (0.42, 0, 1.0, 1.0).
- Ease-out—the ease-out function is equivalent to cubic-bezier (0, 0, 0.58, 1.0).
- Ease-in-out—the ease-in-out function is equivalent to cubic-bezier (0.42, 0, 0.58, 1.0)
- Cubic-bezier—specifies a cubic-bezier curve of which the P0 and P3 points are (0,0) and (1,1), respectively. The four values specify points P1 and P2 of the curve as (x1, y1, x2, y2).

The following example applies a transition effect on the color definition in the P element.

```
p {
  -webkit-transition: color 2s linear;
  font-size: medium;
  font-family: Arial, Helvetica, sans-serif;
  color: #FF0000;
}
p:hover {
  font-family: Arial, Helvetica, sans-serif;
  color: #0000FF;
}
```

As you move over any text using the P element the text will slowly change from red to blue. When you move away from the text it will change back. [Figure 2.35](#) illustrates several paragraphs of text using the P element.

In the figure, the top paragraph is red, the third has transitioned to blue, and the fourth is transitioning from one color to the next. You can elect to have all of the properties be selected as part of the transition by changing the property value to “ALL” as in the following example.

Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt. Neque porro quisquam est qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit, sed quia non numquam eius modi tempora incidunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim ad minima veniam, quis nostrum exercitationem ullam corporis suscipit laboriosam, nisi ut aliquid ex ea commodi consequatur? Quis autem vel eum iure reprehenderit qui in ea voluptate velit esse quam nihil molestiae consequatur, vel illum qui dolorem eum fugiat quo voluptas nulla pariatur.

Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt. Neque porro quisquam est qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit, sed quia non numquam eius modi tempora incidunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim ad minima veniam, quis nostrum exercitationem ullam corporis suscipit laboriosam, nisi ut aliquid ex ea commodi consequatur? Quis autem vel eum iure reprehenderit qui in ea voluptate velit esse quam nihil molestiae consequatur, vel illum qui dolorem eum fugiat quo voluptas nulla pariatur.

Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt. Neque porro quisquam est qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit, sed quia non numquam eius modi tempora incidunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim ad minima veniam, quis nostrum exercitationem ullam corporis suscipit laboriosam, nisi ut aliquid ex ea commodi consequatur? Quis autem vel eum iure reprehenderit qui in ea voluptate velit esse quam nihil molestiae consequatur, vel illum qui dolorem eum fugiat quo voluptas nulla pariatur.

Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt. Neque porro quisquam est qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit, sed quia non numquam eius modi tempora incidunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim ad minima veniam, quis nostrum exercitationem ullam corporis suscipit laboriosam, nisi ut aliquid ex ea commodi consequatur? Quis autem vel eum iure reprehenderit qui in ea voluptate velit esse quam nihil molestiae consequatur, vel illum qui dolorem eum fugiat quo voluptas nulla pariatur.

**Figure 2.35** The transition effect allows you to move simple animation from one state to another.



Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt. Neque porro quisquam est, qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit, sed quia non numquam eius modi tempora incidunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim ad minima veniam, quis nostrum exercitationem ullam corporis suscipit laboriosam, nisi ut aliquid ex ea commodi consequatur? Quis autem vel eum iure reprehenderit qui in ea voluptate velit esse quam nihil molestiae consequatur, vel illum qui dolorem eum fugiat quo voluptas nulla pariatur.

Nemo enim ipsam voluptatem quia voluptas sit aspernatur ad odit et fugit, sed quia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt. Neque porro quisquam est, qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit, sed quia non numquam eius modi tempora incidunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim ad minima veniam, quis nostrum exercitationem ullam corporis suscipit laboriosam, nisi ut aliquid ex ea commodi consequatur? Quis autem vel eum iure reprehenderit qui in ea voluptate velit esse quam nihil molestiae consequatur, vel illum qui dolorem eum fugiat quo voluptas nulla pariatur.

Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt. Neque porro quisquam est, qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit, sed quia non numquam eius modi tempora incidunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim ad minima veniam, quis nostrum exercitationem ullam corporis suscipit laboriosam, nisi ut aliquid ex ea commodi consequatur? Quis autem vel eum iure reprehenderit qui in ea voluptate velit esse quam nihil molestiae consequatur, vel illum qui dolorem eum fugiat quo voluptas nulla pariatur.

```
p {
  -webkit-transition: all 2s linear;
  font-size: medium;
  font-family: Arial, Helvetica, sans-serif;
  color: #FF0000;
}
p:hover {
  font-family: Arial, Helvetica, sans-serif;
  font-size: xx-large;
  color: #0000FF;
}
```

For quick, simple animation sequences, transitions are great.

For more complex animation you will want to use the new transform settings. The following HTML and CSS style allows you to add a bouncing text block to the screen.

```
<html>  
  <head>  
    <title>Bouncing Box example</title>  
    <style type="text/css" media="screen">  
      @-webkit-keyframes bounce {  
        from {  
          left: 0px;  
        }  
      }
```

```
        to {
            left: 400px;
        }
    }
    .animation {
        -webkit-animation-name: bounce;
        -webkit-animation-duration: 2s;
        -webkit-animation-iteration-count: 4;
        -webkit-animation-direction: alternate;
        position: relative;
        left: 0px;
    }
</style>
</head>
<body>
    <p class="animation">
        The text bounces back and forth
    </p>
</body>
</html>
```

The animation is controlled through the use of the style sheet. There are two parts you need to control. The first sets up the type of animation you want to use. Here the setting is for an animation sequence named `bounce`. The animation and the movement will be from 0 px to the left 400 px.

```
@-webkit-keyframes bounce {
    from {
        left: 0px;
    }
    to {
        left: 400px;
    }
}
```

The next step is to define what gets animated. In this example you have a CSS class associated with the `bounce` animation. There are a couple of additional settings. The `duration` setting controls how long each animation sequence takes to play in seconds, and the `count` setting specifies how many times the animation plays. Together, it looks as follows.

```
.animation {
    -webkit-animation-name: bounce;
    -webkit-animation-duration: 2s;
    -webkit-animation-iteration-count: 4;
    -webkit-animation-direction: alternate;
    position: relative;
    left: 0px;
```

Currently, the examples above will only work in the latest versions of Safari and Google's Chrome. If, however, you have an iPhone or iPod Touch then your version of Safari already supports the new CSS animation sequences.

## Delivering Solutions for the Mobile Market

Waiting for PC computers to catch up and support HTML5 may be eclipsed by the rapid adoption of smart phone and compact mobile devices spilling onto the market.

Smart mobile phones are receiving a lot of attention from the sheer power they pack. This power is extended to the mobile web browsers installed on these devices. The popular Apple iPhone runs Mobile Safari, a browser built from the Open Source WebKit project. Google's Android mobile OS and Palm's Pre WebOS are also built from WebKit. Not to be out done, Mozilla and Opera have mobile browsers, too. All of these browsers run HTML5.

The problem is screen size. The real estate space for a Windows 7 PC can be ten times greater than the humble  $480 \times 320$  space of the iPhone. To help you, CSS3 has a final trick up its sleeve.

The media definition in CSS allows you to identify different styles for different media types. Originally defined in CSS2, the CSS3 expands the functionality of the CSS2 version to allow you to specify any type of device.

The easiest place to use the media definition is right when you link to a CSS document in the head of the web page. Typically you will write the following code to link to a CSS document.

```
<link rel="stylesheet" type="text/css"
href="style.css">
```

The media definition now allows you to specify a style to be associated with a device. Take, for instance, the following CSS link reference to two styles documents.

```
<link rel="stylesheet" type="text/css"
media="screen" href="screen.css">
<link rel="stylesheet" type="text/css"
media="print" href="print.css">
```

The first link uses the media definition to target a CSS document from the computer screen. The second CSS document targets how data are presented when the document is printed. Using this technique you can create two different presentation styles using the same content. One style is used for screen presentation and the other for print. Below is a list of the media names you can use:

- All—suitable for all devices
- Braille—intended for braille tactile feedback devices

- Embossed—intended for paged braille printers
- Handheld—intended for handheld devices (typically small screen, limited bandwidth)
- Print—intended for paged material and for documents viewed on screen in print preview mode
- Projection—intended for projected presentations (e.g., projectors)
- Screen—intended primarily for color computer screens
- Speech—intended for speech synthesizers
- tty—intended for media using a fixed-pitch character grid (such as teletypes, terminals, or portable devices with limited display capabilities)
- tv—intended for television-type devices (low resolution, color, limited-scrollability screens, sound available)

Having the names is great, but it does not help when there are so many different devices coming on to the market with different screen resolutions. To help with this, you can modify the media type to look for screen resolutions and deliver the appropriate style sheet. Using the property device-width you can specify a style sheet for a specific width.

```
<link rel="stylesheet" type="text/css"
media="(device-width: 3200px)" href="iphone.css">
```

Using CSS you can dynamically change the presentation of the content to best suite the device accessing the content.

## What You Have Learned

CSS3 is an amazing advancement for Cascading Style Sheets. In this chapter you have seen how you have absolute control over your design using CSS to control placement of elements on the screen, the font structure, measurement, and color. CSS3 extends further from earlier versions of CSS to include basic and rich animation techniques and media management tools. Of all the technologies in HTML5, CSS is arguably receiving the most attention. The latest standards for CSS3 can be found at <http://www.w3.org/Style/CSS/current-work>.

# focal press books

## VISUALIZING THE WEB WITH HTML 5

Download these other great articles in the HTML5: Visualizing the Web series

### HTML5 Rich Media Foundation

Learn about the new ways in which video and audio can be easily embedded into your HTML5 Web pages. Discover how you can create new Web media content and how JavaScript, CSS, and SVG can be integrated to create a compelling, rich media foundation for your work.

### Rendering HTML5 Illustration

Scalable Vector Graphics (SVG) and CANVAS are two new tools introduced in HTML5 that you can use to add illustration and interactive animation to your Web pages. Understand why there are two different formats, how you can use them, and where they can be used today.

### Picture CSS3

See how you can use Cascading Style Sheets 3, or CSS3, to quickly and easily increase your control over visual Web page design.

### The HTML5 JavaScript Model

JavaScript is the glue that enables HTML to become interactive. Learn how you can leverage JavaScript Libraries to quickly build beautiful Web applications.

### HTML5 Tag Structure

Take a gander at the new HTML5 elements: how you can take advantage of them and what you need to look out for as you design your new Web sites.

And look for the HTML5: Visualizing the Web book coming in July 2010.

The book will feature information and resources that expand on the 5-part series.

learn • master • create



Visit [focalpress.com](http://focalpress.com) for video tutorials, sample chapters, podcasts and exclusive competitions and discounts.

f o c a l p r e s s . c o m